HOW TO CONFIGURE ROUTER ON A STICK - 802.1Q TRUNK TO CISCO ROUTER

WRITTEN BY ADMINISTRATOR. POSTED IN [CISCO ROUTERS - CONFIGURING CISCO ROUTERS](http://www.firewall.cx/cisco-technical-knowledgebase/cisco-routers.html)

4.118644067811111 Rating 4.12 (59 Votes)

inShare

Router-on-a-stick is a term frequently used to describe a setup up that consists of a router and switch connected using one Ethernet link configured as an 802.1q trunk link. In this setup, the switch is configured with multiple VLANs and the router performs all routing between the different networks/VLANs.

While some believe the term 'router-on-a-stick' sounds a bit silly, it's a very popular term and commonly used in networks where no layer-3 switch exists. A good example of a router-on-a-stick configuration (which also happens to be the one we are going to cover) would be a Call Manager Express installation where there is the need to split the VoIP network, consisting of your Cisco IP Phone devices, from your data network where all workstations and servers are located.

EXAMPLE SCENARIO

Our example is based on a scenario you are most likely to come across when dealing with VoIP networks. Because VoIP implementations require you to separate the data and voice network in order to route packets between them, you need either a layer 3 switch or a router. This configuration ensures availability and stability of the VoIP service, especially during peak traffic hours in your network.

Packets running between VLANs are routed via the CCME router connected to the switch using one physical port configured as a trunk port on both ends (switch and router). If you would like to read more on VLAN routing and VLAN theory, you can visit our popular [VLAN Section](http://www.firewall.cx/networking-topics/vlan-networks.html) that covers all related topics and terms found in this article.

This example will show you how to configure a Cisco router and switch in order to create a trunk link between them and have the router route packets between your VLANs.

This diagram shows an illustration of the above configuration.



STEP 1 - SWITCH CONFIGURATION

First step is to create the required two VLANs on our Cisco switch and configure them with an IP address. Since all Cisco swiches contain VLAN1 (Default VLAN), we only need to create VLAN2:

SW1# **configure terminal**
SW1(config)# **vlan2**
SW1(config-vlan)# **name voice**
SW1(config-vlan)# **exit**SW1(config)# **interface vlan1**
SW1(config-if)# **description Data Vlan**
SW1(config-if)# **ip address 192.168.0.2 255.255.255.0**
SW1(config-if)# **exit**
SW1(config)# **interface vlan2**
SW1(config-if)# **description Voice Vlan**
SW1(config-if)#**ip address 192.168.2.2 255.255.255.0**
SW1(config-if)# **exit**

Next, we need to create the trunk port that will connect to the router. For this purpose, we've selected port GigabitEthernet 0/1 (port 1):

SW1# **configure terminal**
SW1(config)# **interface gigabitethernet 0/1**
SW1(config-if)# **description Trunk-to-Router**
SW1(config-if)# **switchport trunk encapsulation dot1q**
SW1(config-if)# **switchport mode trunk**
SW1(config-if)# **spanning-tree portfast trunk**

To eliminate confusion, these commands are instructing the switch thus:

1) Define the trunk to use the 802.1q protocol

2) Set the specific port to **trunk mode**

3) Enable the **spanning-tree** **portfast trunk** function to ensure the port will forward packets immediately when connected to a device e.g router. Note: The **spanning-tree** **portfast trunk**command should never be used on ports that connect to another switch, to ensure network loops are avoided.

The above steps complete the switch-side configuration.

STEP 2 - ROUTER CONFIGURATION

We need to follow a similar configuration for our router to enable communication with our switch and allow all VLAN traffic to pass through and route as necessary.

Creating a trunk link on a router port is not very different from the process used above - while we create the trunk port on one physical interface, we are required to create a sub-interface for each VLAN.

Again, this is a fairly simple process and easy to understand once you've done it at least one time.

R1# **configure terminal**
R1(config)# **interface gigabitethernet0/1**
R1(config-if)# **no ip address**
R1(config-if)# **duplex auto**
R1(config-if)# **speed auto**
R1(config-if)# **interface gigabitethernet0/1.1**
R1(config-subif)# **description Data VLAN**
R1(config-subif)# **encapsulation dot1q 1 native**
R1(config-subif)#**ip address 192.168.0.1 255.255.255.0**
R1(config-subif)# **ip nat inside**
R1(config-subif)# **ip virtual-reassembly**
R1(config-subif)# **interface gigabitethernet0/1.2**
R1(config-subif)# **description Voice VLAN**
R1(config-subif)# **encapsulation dot1q 2**
R1(config-subif)# **ip address 192.168.2.1 255.255.255.0**
R1(config-subif)# **ip nat inside**
R1(config-subif)# **ip virtual-reassembly**

In order to form a trunk link with our switch it is necessary to create one sub-interface for every VLAN configured on our switch. After creating the sub-interface, we assign an IP address to it and set the encapsulation type to 802.1q along with the VLAN to which the subinterface belongs.

For example, the **encapsulation dot1q 2** command defines 802.1q encapsulation and sets the subinterface to VLAN 2. The **native** parameter we used for subinterface **gigabitethernet0/1.1** tells the router that the native vlan is VLAN 1. This is a default parameter on every Cisco switch and therefore must be matched by the router as well.

The **ip virtual-reassembly** command is usually automatically thrown in by the Cisco IOS (we've included it to show you the command) and is a security measure to avoid buffer overflow and control memory usage during an attack of fragmented packets which can cough up your router's resources. This command is added automatically when you enable the NAT service using the **ip nat inside** command. More information on NAT configuration can be obtained by our Cisco Router NAT Configuration articles.

ARTICLE SUMMARY

This article explained the use of router-on-a-stick configurations and showed how you can configure an 802.1q trunk link between a Cisco switch and router. Router-on-a-stick configurations are extremely useful in environments where no layer-3 switch exists, providing Inter-VLAN routing services with a single router and one interface - cutting down seriously the costs for internal routing.

It is always preferable to use a router with a Gigabit Ethernet interface to ensure you've got plenty of bandwidth to handle large amounts of data transfers if needed.

If you have found the article useful, we would really appreciate you sharing it with others by using the services provided at the top left corner of this article. Sharing our articles takes only a minute of your time and helps Firewall.cx reach more people through such services.

# <http://www.firewall.cx/cisco-technical-knowledgebase/cisco-routers/812-cisco-router-dhcp-config.html>

# HOW TO CONFIGURE DHCP SERVER ON A CISCO ROUTER

WRITTEN BY ADMINISTRATOR. POSTED IN [CISCO ROUTERS - CONFIGURING CISCO ROUTERS](http://www.firewall.cx/cisco-technical-knowledgebase/cisco-routers.html)

4.3584905660411111 Rating 4.36 (53 Votes)

inShare

DHCP (Dynamic Host Configuration Protocol) is the protocol used by network devices (such as PCs, network printers, etc) to automatically obtain correct network parameters so they can access network and Internet resources such as IP Address, Default Gateway, Domain Name, DNS Servers and more.

A DHCP Server is considered necessary in today's networks. Devices usally found providing this service are Windows servers, routers and layer 3 switches.

This article describes how to configure basic DHCP parameters on a Cisco router, enabling it to act as a DHCP server for your network.

Users should also read our article [Resolving Cisco Switch & Router ‘DHCP Server Pool Exhausted-Empty’ Error – Client IP Assignment Failure](http://www.firewall.cx/cisco-technical-knowledgebase/cisco-switches/1063-cisco-switch-router-dhcp-server-conflicts.html) to help them deal with the most common problem that arises when the DHCP server stops serving IP addresses to network clients.

#### EXAMPLE SCENARIO

For the sake of this article, suppose we have the network shown in the following diagram, for which we would like to enable the DHCP service on our Cisco router.

The router will act as a DHCP server for the 192.168.1.0/24 network. IP Addresses already assigned to our switch (192.168.1.2) and File Server (192.168.1.5) will be excluded from the DHCP pool, to ensure they are not given out to other hosts and cause an IP address conflict.

First step is to enable the DHCP service on our router, which by default is enabled.



First step is to enable the DHCP service on our router, which by default is enabled:

R1# **configure terminal**
R1(config)# **service dhcp**

Next step is to create the DHCP pool that defines the network of IP addresses that will be given out to the clients. Note that 'NET-POOL' is the name of the DHCP IP Pool we are creating:

R1(config)# **ip dhcp pool NET-POOL**
R1(dhcp-config)# **network 192.168.1.0 255.255.255.0**

This tells the router to issue IP addresses for the network 192.168.1.0, which translates to the range 192.168.1.1 - 192.168.1.254. We will have to exclude the IP addresses we want later on.

We now define the DHCP parameters that will be given to each client. These include the default gateway (default-router), dns servers, domain and lease period (days):

R1(dhcp-config)# **default-router 192.168.1.1**
R1(dhcp-config)# **dns-server 192.168.1.5 195.170.0.1**
R1(dhcp-config)# **domain-name Firewall.cx**
R1(dhcp-config)# **lease 9**

 The **domain-name** and **lease** parameters are not mandatory. By default, the **lease time** for an IP address is one day, however we can specify any time range we need. For example, if we need to set the lease time for **4 hours** and **30 minutes** we would use the following command under our DHCP pool:

R1(dhcp-config)#**lease 0 4 30**

The above command is interpreted as follows: **0** (**Zero**) days, **4** hours and **30** minutes.

All we need now is to exclude the IP addresses we don't want our DHCP server giving out. Drop back to 'global configuration mode' and enter the following:

R1(config)# **ip dhcp excluded-address 192.168.1.1 192.168.1.5**
R1(config)#**ip dhcp excluded-address 192.168.1.10**

This excludes IP addresses 192.168.1.1 - 192.168.1.5 & 192.168.1.10. As you can see, there's an option to exclude a range of IP addresses or a specific address.

The above configuration is all you need to get the DHCP server running for your network. We'll provide a few more commands you can use to troubleshoot and ensure it's working correctly.

The following command will allow you to check which clients have been served by the DHCP:

R1# **show ip dhcp binding**
Bindings from all pools not associated with VRF:
IP address Client-ID/ Lease expiration Type
Hardware address/
User name
192.168.1.6 0100.1e7a.c409 Jan 19 2009 03:06 PM Automatic
192.168.1.7 0100.1e7a.c3c1 Jan 19 2009 09:00 PM Automatic
192.168.1.8 0100.1ebe.923b Jan 19 2009 02:25 PM Automatic
192.168.1.9 0100.1b53.5ccc Jan 19 2009 02:03 PM Automatic
192.168.1.11 0100.1e7a.261d Jan 19 2009 07:52 PM Automatic
R1#

Notice that IP addresses 192.168.1.5 & 192.168.1.10 have not been assigned to the clients.

#### ARTICLE SUMMARY

In this article we've covered how a Cisco router can be used as a basic DHCP server and the various options available. We also saw how you can obtain general information about the service. There are more options available with the DHCP service, however this basic article should cover most of your network needs.

Future DHCP articles will explore advanced options and debugging for more complex networks containing VLANs and IP Telephony.

If you have found the article useful, we would really appreciate you sharing it with others by using the provided services on the top left corner of this article. Sharing our articles takes only a minute of your time and helps Firewall.cx reach more people through such services.

#### ABOUT THE WRITER

Ammar Muqaddas is a CCNA certified Engineer, CCNA Instructor and member of the Firewall.cx Team.